OIPE JC46
JUL 18 2005
PATENT & TRADEMARK OFFICE

AF/IFW

TRANSMITTAL FORM	Attorney Docket No. CA920000033US1/2033P

In re the application: **Iqbal A. GORALWALLA et al.** Confirmation No: **3709**

Serial No: **09/872,827** Group Art Unit: **2167**

Filed: **May 31, 2001** Examiner: **Wong, Leslie**

For: **Optimizing Updatable Scrollable Cursors In Database Systems**

ENCLOSURES ***(check all that apply)***

□	Amendment/Reply	□	Assignment and Recordation Cover Sheet	□	After Allowance Communication to Group
	□ After Final	□	Part B-Issue Fee Transmittal	□	Notice of Appeal
□	Information disclosure statement	□	Letter to Draftsman	■	Appeal Brief
	□ Form 1449	□	Drawings	□	Status Letter
	□ (X) Copies of References	□	Petition	■	Postcard
□	Extension of Time Request *	□	Fee Address Indication Form	□	Other Enclosure(s) *(please identify below):*
□	Express Abandonment	□	Terminal Disclaimer		
□	Certified Copy of Priority Doc	□	Power of Attorney and Revocation of Prior Powers		
□	Response to Incomplete Appln	□	Change of Correspondence Address		
□	Response to Missing Parts	*Extension of Term: Pursuant to 37 CFR 1.136, Applicant petitions the Commissioner to extend the time for response for xxxxxx month(s), from to .			
	□ Executed Declaration by Inventor(s)				

CLAIMS

FOR	Claims Remaining After Amendment	Highest # of Claims Previously Paid For	Extra Claims	RATE	FEE
Total Claims	24	24	0	$ 50.00	$ 0.00
Independent Claims	6	6	0	$200.00	$ 0.00
				Total Fees	$ 0.00

METHOD OF PAYMENT

□ Check no. ______ in the amount of $ __________ is enclosed for payment of fees.

■ Charge **$500.00** to Deposit Account No. 09-0460 (IBM Corporation) for payment of fees.

■ Charge any additional fees or credit any overpayment to Deposit Account No. 09-0460 (IBM Corporation).

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Attorney Name	Joyce Tom, Reg. No. 48,681
Signature	
Date	July 14, 2005

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on **July 14, 2005**

Type or printed name	Jinny Nguyen
Signature	Jnguyen

OIPE JC46
JUL 1 8 2005
PATENT & TRADEMARK OFFICE

CERTIFICATE OF MAIL

I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class Mail in an envelope addressed to Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on **July 14, 2005**.

Jinny Nguyen

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

APPEAL NO:

In Re Application of:	Date: July 14, 2005
Iqbal A. GORALWALLA et al.	Confirmation No.: 3709
Serial No: 09/872,827	Group Art Unit: 2167
Filed: May 31, 2001	Examiner: Wong, Leslie

For: OPTIMIZING UPDATABLE SCROLLABLE CURSORS IN DATABASE SYSTEMS

APPEAL BRIEF

07/18/2005 HVUONG1 00000056 090460 09872827
01 FC:1402 500.00 DA

Joyce Tom
Attorney for Appellant
Sawyer Law Group, LLP
2465 E. Bayshore Road, Suite 406
Palo Alto, CA 94303

O I P E JC46
JUL 1 8 2005
PATENT & TRADEMARK OFFICE

TABLE OF CONTENTS

OIPE JC46
JUL 1 8 2005
PATENT & TRADEMARK OFFICE

I. REAL PARTY IN INTEREST

Appellants respectfully submit that International Business Machines Corporation is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellants state that no such proceeding exists.

III. STATUS OF CLAIMS

Claims 1-24 are pending and stand rejected. Claim 9 was amended to correct a typographical error, and claims 22-24 were added. Accordingly, claims 1-24 are on appeal and all applied rejections concerning those claims are herein being appealed.

IV. STATUS OF AMENDMENT

All amendments have been entered.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present invention is directed to optimizing command execution in a relational database management system (RDBMS) where data records are stored in data pages. According to the present invention, *each* data page, not just dirty data pages, is provided with an identifier which indicates when the page was modified last. (Specification, page 3, lines 12-18). Thus, whenever a record on the data page is modified, the identifier necessarily changes. When the RDBMS receives a command to access a data record on a data page, such as an UPDATE or DELETE command, the system copies the selected *data record* from the page, along with the page's identifier, and stores them temporarily in a secondary storage area. (Specification, page 3, line 19 to page 4, line 3).

The system uses the stored identifier to verify that the copied record has not been modified during the period in which it was copied and stored. For instance, the system compares the stored identifier with a current identifier for the page. If the page has *not been modified*, the stored identifier and the current identifier will be the same, and the system can conclude that the stored data record matches the current record in the database. Accordingly, when the stored identifier is the same as the current identifier, the system is not required to compare the data record in the temporary copy with the record in the current table, which can be a costly event if the records are extensive. (Specification, page 5, lines 1-11).

By providing *each data page* with the identifier and then storing the identifier with the data record in the secondary storage, the RDBMS can quickly and easily determine whether the data page containing the stored data record was modified by comparing identifiers. Accordingly, the present invention minimizes the number of times the RDBMS must go into the data page itself to verify that the stored record is consistent with the current record.

In independent claims 1 and 9 are a method and programming instructions for optimizing command execution in a database system that stores data records on a plurality of data pages is provided. Referring to FIG. 1 of the Specification, the method includes providing an identifier 22 to each data page 10. The identifier 22 indicates when any of the data records 12, 14 contained in the page 10 were last modified (Spec., page 3, line 13-16). The method also includes selecting a data record 12 from the data page 10 and copying the selected data record 12 to a second storage area 16 (Spec., page 3, lines 19-22). The method also includes verifying that the selected data record 12 has not been modified since the time that it was copied to the second storage area 16 based upon the identifier 22 (Spec., page 5, lines 1-3), and executing the command.

Independent claim 17 is directed to a database management system comprising data records 12, 14 stored on a plurality of data pages 10, means for providing an identifier 22 on each data page 10, means for selecting a data record 12 from a data page 10, means for copying and storing the selected data record 12 and the identifier 22 from the data page to a second storage area 16, means for determining a current identifier 22 from the data page 10 and means for verifying that the selected data record 12 has not been modified since the time that it was copied to the second storage area 16 by determining that the stored identifier 24 is the same as the current identifier 22 from the data page 10.

Independent claims 22 and 23 are directed to a method and programming instructions for optimizing command execution in a database system, and independent claim 24 is directed to a database management system comprising data records 12, 14 stored on a plurality of data pages

10. Claims 22 and 23 include providing an identifier 22 to each data page 10 where the identifier 22 indicates when any of the data records 12, 14 were last modified, selecting a data record 12 from a data page 10, and copying the selected data record 12 and copying and storing the identifier 22 to a second storage area 16. The method and programming instructions also include verifying that the selected data record 12 has not been modified since the time that it was copied to the second storage area 16 based upon the identifier 22 by determining a current identifier 22 for the data page 10, comparing the current identifier 22 with the stored identifier 24, and concluding the selected data record 12 has not been modified when the current identifier 22 is the same as the stored identifier 24. The database management system of claim 24 includes means for performing the method of claim 22.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The Examiner rejected claims 1-24 under 35 U.S.C. §103(a) as being unpatentable over Josten et al. (U.S. Patent No. 5,574,902) in view of Ponnekanti (U.S. Patent No. 6,591,269).

VII. ARGUMENTS

A. Summary of the Applied Rejections

In the Final Office Action, the Examiner rejected claims 1-21 under 35 U.S.C. §103(a) as being unpatentable over Josten et al. (U.S. Patent No. 5,574,902) in view of Ponnekanti (U.S. Patent No. 6,591,269). In so doing, the Examiner stated:

> Regarding claim 1, Josten teaches a method for optimizing command execution in a database system, wherein data records are stored on a plurality of data pages therein (col. 5, line 65-col. 6, line 6), the method comprising the steps of:
>
> (a) 'providing an identifier to each data page' as an ordinal number (ORD#) is assigned to a data page buffer control block (BCB) in dirty page list (DPL) (col. 7, lines 10-16 and 42-52; col. 11, lines 44-46, and Fig. 2, element 44 (i.e., ordinal #));
>
> (b) 'selecting a data record from a data page' as list 42 includes a series of the ordinal number from DPL that are associated with data pages in the LCB that were accessed by the transaction corresponding to TPL-1 (col. 7, line 53 – col. 8, line 2; col. 5, lines 5-10; col. 7, lines 10-16; col. 8, lines 1-7; col. 8, lines 46-51);
>
> (c) 'copying the selected data record to a second storage area' as the data manager issues a SETWRITE request to indicate intent to update the named data page (col. 7, lines 10-16; col. 8, lines 9-18);
>
> (d) 'verifying that the selected data record has not been modified since the time that it was copied to the second storage area based upon the identifier' as a test is made to detect a consecutive update to the same data page by comparing the ORD# of the last entry of the transaction page list (TPL) with the ORD# of the new entry in the buffer control block (col. 11, lines 28-38); and
>
> (e) 'executing the command' as committing transactions schedules write I/Os for all TPL entries (col. 18, lines 3-7).
>
> a). Josten does not explicitly teach the identifier indicating when any of the data records contained therein were last modified.
>
> Ponnekanti, however, teaches the log records contain only the PAGEIDs and the timestamps of the source page and the target page and the positions of the first and last key that were copied (col. 11, lines 47-49).
>
> It would have been obvious to one of ordinary skill in the art at the time of the invention was made to combine the teachings of the cited references

because Ponnekanti's teaching would have allowed Josten's to enhance the speed in which the database server stores, retrieves, and processes particular data records as indicated by Ponnekanti at col. 3, lines 1-10 and col. 19, lines 40-50.

Regarding claims 2 and 10, Josten further teaches wherein the copying step (c) includes:

(c1) copying and storing the identifier to the second storage area (col. 4 lines 27-32 and col. 7, line 57-col. 8, line 2).

Regarding claims 3 and 11, Josten further teaches wherein the verifying step (d) includes:

(d1) determining a current identifier for the data page (col. 11, lines 28-31);

(d2) comparing the current identifier with the stored identifier (col. 11, lines 34-36); and

(d3) concluding the selected data record has not been modified when the current identifier is the same as the stored identifier (col. 11, lines 36-38).

B. The Cited Prior Art

Josten is directed to a method for destaging updated locally cache pages in a multisystem shared disk environment, such as a Sysplex, with a high-speed shared electronic store. In Josten, each computer processing complex (CPC) caches data pages in its local cache buffer (LCB). When one of those pages is updated, it eventually must be written to shared stable storage ("externalized") and cross-invalidated in other CPCs when a transaction commits. (Col. 4, lines 8-13). Josten presents (a) an efficient process for identifying the set of cached pages that must be externalized and (b) a fast process for scheduling of the externalizing write I/Os so that the corresponding transaction can release its locks. (Col. 4, lines 13-18). To do this, Josten assigns an ordinal number (ORD#) to a *dirty data page*, which is defined as a page that contains a record that has been updated by a transaction, but has not yet been written to external stable storage. Josten then places the ORD# in a list associated with the transaction (transaction page list (TPL)). (Col. 7, lines 1-18).

Ponnekanti is directed to performing an online rebuild of a B+ tree index by copying the index rows to newly allocated pages in the key order so that good space utilization and clustering are achieved. (Abstract). In Ponnekanti, index rows are copied to newly allocated pages in the key order, and old pages are deallocated. Multiple leaf pages are rebuilt and then changes to higher levels are propagated. While propagating leaf level changes to higher levels, level 1 pages are reorganized. (Abstract). Thus, the need for a separate pass is eliminated.

C.. Independent Claims 1, 9 and 17 Are Allowable Over the Cited References.

Appellants respectfully submit that Josten and Ponnekanti fail to teach or suggest "providing an identifier to *each* data page" in a database system, as recited in claims 1 and 9, and "means for providing an identifier on *each* data page" in the database system, as recited in claim 17, where the identifier indicates "when any of the data records contained therein were last modified."

According to the Final Office Action, the Examiner suggests that the identifier of the present invention is taught by Josten's ORD# which is assigned to a *dirty* data page in the local cache buffer (LCB). (Col. 7, lines 10-16). Thus, a clean data page is not assigned an ORD#. In the present invention, the identifier is provided to *each data page*, regardless of whether the page is clean or dirty. Accordingly, Appellants respectfully submit that Josten's ORD# is not equivalent to the identifier of the present invention.

Moreover, the identifier indicates *when* any of the data records contained in the data page was modified last. The Examiner concedes that the ORD# does not provide such an indication, but suggests that Ponnekanti's log records do at column 11, lines 47-49. Appellants respectfully disagree. In Ponnekanti, the application runs as a sequence of transactions, with each transaction rebuilding leaf pages in the page chain. At the end of each transaction, the new pages generated are flushed to disk and then the old pages that were removed from the tree are made available for fresh allocations. The log records contain PAGEIDs and timestamps of the source page and the target page, as well as positions of a first and a last key copied. (Column 11, lines 30-49).

The PAGEID and timestamp of a source or target page has no relation to Josten's ORD#. In addition, nothing in Ponnekanti teaches or suggests that the timestamp of the source or target page indicates "when any of the data records contained therein were last modified," as recited in claims 1, 9 and 17. Accordingly, Appellants respectfully submit that the combination of Josten and Ponnetanki fail to teach or suggest the identifier of the present invention, as recited in claims 1, 9 and 17.

In addition, Appellants respectfully maintain that neither reference teaches or suggests "verifying that the selected data record has not been modified since the time that it was copied to the second storage area *based upon the identifier*," as recited in claims 1 and 9, and "means for verifying that the selected data record has not been modified since the time that it was copied to the second storage area by determining that the stored identifier is the same as the current identifier from the data page," as recited in claim 17. In Josten, the ORD# for a dirty page does not change once the page becomes dirty. If a transaction updates a data page *more than once*, the ORD# for the dirty data page does not change. Thus, in Josten, it is not possible to verify that "the selected data record has not been modified since the time that it was copied to the second storage area *based upon the identifier*," as recited in claims 1 and 9, and Josten does not teach or suggest "means for verifying that the selected data record has not been modified since the time that it was copied to the second storage area by determining that the stored identifier is the same as the current identifier from the data page," as recited in claim 17.

In the Final Office Action, the Examiner states that this feature is taught by Josten at column 11, lines 28-38. Appellants respectfully disagree. In the cited portion, Josten discusses avoiding duplicate TPL entries by determining whether a consecutive update has been performed on the same data page by comparing the ORD# of the last entry of the TPL with the ORD# of the new entry. If the ORD#s are *different*, the new entry is made at the end of the TPL. This process does not verify "that the selected data record has not been modified since the time that it was copied to the second storage area," as recited in claims 1, 9 and 17.

For the foregoing reasons, Appellants respectfully submit that claims 1, 9 and 17 are allowable over Josten and Ponnekanti. Claims 2-8, 10-16, and 18-21 depend on claims 1, 9 and 17, respectively, and therefore, the above arguments apply with full force. Thus, claims 2-8, 10-16, and 18-21 are also allowable over Josten in view of Ponnekanti.

D. Independent Claims 22, 23 and 24 Are Allowable Over the Cited References.

Claim 22 incorporates claims 1, 2 and 3, claim 23 incorporates claims 9, 10 and 11, and claim 24 is a system claim consistent with claims 22 and 23. As such, claims 22, 23 and 24 are allowable over Josten and Ponnekanti for the reasons set forth above. In addition, claims 22-24 are allowable because neither of the references teaches or suggests steps (d1), (d2) and (d3). In the Final Office Action, the Examiner asserts that Josten teaches steps (d1)-(d3) at column 11, lines 28-38. The cited portion states,

> Efficient tracking of data pages in a data base that are modified by a transaction is provided by the transaction page list (TPL) of this invention, each entry of which consists of the associated ORD # only. Before adding an entry to the TPL for a transaction, a simple test is made to detect a consecutive update to the same data page by comparing the ORD # of the last entry of the TPL with the ORD # of the new entry, which is available in the buffer-control block (BCB) for the data page copy in LCB. If the last TPL entry is not the same as the entry for ORD #, the new entry is made at the end of the TPL. Consecutive duplicate TPL

> entries are thereby avoided, which reduces the TPL size without significant TPL search activity.

In the advisory action, the Examiner states:

> Josten teaches 'verifying that the selected data record has not been modified since the time that it was copied to the second storage area based upon the identifier' as a test is made to detect a consecutive update to the same data page by comparing the ORD # of the last entry of the transaction page list (TPL) with the ORD # of the new entry in the buffer control block (col. 11, lines 28-38) as claimed.

Appellants disagree.

In Josten, if a clean data page becomes dirty (i.e., it's modified by a first transaction), an ORD# is assigned to the dirty page, and the ORD# is stored in the dirty page list (DPL). According to Josten, "[t]he ORD# associated with a data page in LCB remains the same until the data page is written to stable storage." Column 11, lines 24-25. Thus, once a clean page becomes dirty, it remains dirty until it is externalized. Subsequent modifications to the same dirty page do not change the dirty page's ORD#.

The portion cited above describes how the size of the TPL can be reduced by eliminating consecutive duplicate TPL entries. As stated in Josten, each TPL entry "consists of the associated ORD# only." (Column 11, line 31). If two consecutive transactions operate on the same data page, the ORD# for the first transaction is the same ORD# as the second transaction. Without Josten's "test," the TPL would have two consecutive entries that are duplicates. With the "test," consecutive duplicate TPL entries are avoided.

In the present invention, the current identifier for the data page is compared to the stored identifier for the data page. If the current identifier is the **same** as the stored identifier, the present invention concludes that the data record **has not been modified**. In the present invention, this conclusion is logical because the identifier for the data page changes if one of the data records in the data page has been modified. If the identifier for the data page is the same, none of the records has been modified.

In contrast, Josten's ORD# for a data page "remains the same" regardless of the number of times the data page has been modified. If two consecutive transactions modify the same page, the ORD#'s for the first and second transaction are the same. Thus, if the second ORD# (current identifier) is the same as the first ORD# (stored identifier), Josten concludes that the second transaction **has modified** the same data page. This is in direct contrast to the present invention which teaches that the data record in a data page **has not been modified** when the identifiers are the same.

Based on a close comparison of Josten to the present invention, Appellants submit that claims 22, 23 and 24 are allowable over Josten in view of Ponnekanti for this additional and alternative reason.

E. Summary of Arguments

For the reasons set forth above, Appellants respectfully submit that the claims 1-24 are allowable over the cited references. Appellants respectfully request that the final rejection of claims 1-24 be reversed.

Respectfully submitted,
SAWYER LAW GROUP LLP

July 14, 2005
Date

Joyce Tom
Attorney for Appellant
Reg. No. 48,681
(650) 493-4540

O I P E
JUL 1 8 2005
JC46
PATENT & TRADEMARK OFFICE

APPENDIX A

CLAIMS

1. (Original) A method for optimizing command execution in a database system, wherein data records are stored on a plurality of data pages therein, the method comprising the steps of:

(a) providing an identifier to each data page, the identifier indicating when any of the data records contained therein were last modified;

(b) selecting a data record from a data page;

(c) copying the selected data record to a second storage area;

(d) verifying that the selected data record has not been modified since the time that it was copied to the second storage area based upon the identifier; and

(e) executing the command.

2. (Original) The method of claim 1, wherein the copying step (c) includes:

(c1) copying and storing the identifier to the second storage area.

3. (Original) The method of claim 2, wherein the verifying step (d) includes:

(d1) determining a current identifier for the data page;

(d2) comparing the current identifier with the stored identifier; and

(d3) concluding the selected data record has not been modified when the current identifier is the same as the stored identifier.

4. (Original) The method of claim 3, wherein the verifying step (d) further includes:

(d4) determining whether the selected data record has not been modified when the current identifier is not the same as the stored identifier by:

(d4a) accessing a current version of the selected data record on the data page; and

(d4b) comparing the selected data record with the current version of the selected data record.

5. (Original) The method of claim 4, wherein the identifier comprises a time stamp.

6. (Original) The method of claim 4, wherein the identifier comprises a log sequence number (LSN).

7. (Original) The method of claim 1, wherein the second storage area is a temporary data record in a temporary table.

8. (Original) The method of claim 1, wherein the command is a positioned UPDATE and DELETE command in a relational database system supporting scrollable cursors and optimistic concurrency.

9. (Currently amended) A computer readable medium containing programming instructions for optimizing command execution in a database system, wherein data records are stored on a plurality of data pages therein, the programming instructions for:

(a) providing an identifier to each data page, the identifier indicating when any of the data records contained therein were last modified;

(b) selecting a data record from a data page;

(c) copying the selected data record to a second storage area;

(d) verifying that the selected data record has not been modified since the time that it was copied to the second storage area based upon the identifier; and

(e) executing the command.

10. (Original) The computer readable medium of claim 8, wherein the copying instruction (c) includes:

(c1) copying and storing the identifier to the second storage area.

11. (Original) The computer readable medium of claim 9, wherein the verifying instruction (d) includes:

(d1) determining a current identifier for the data page;

(d2) comparing the current identifier with the stored identifier; and

(d3) concluding the selected data record has not been modified when the current identifier is the same as the stored identifier.

12. (Original) The computer readable medium of claim 10, wherein the verifying instruction (d) further includes:

(d4) determining whether the selected data record has not been modified when the current identifier is not the same as the stored identifier by:

(d4a) accessing a current version of the selected data record on the data page; and (d4b) comparing the selected data record with the current version of the selected data record.

13. (Original) The computer readable medium of claim 12, wherein the identifier comprises a log sequence number (LSN).

14. (Original) The computer readable medium of claim 12, wherein the identifier comprises a time stamp.

15. (Original) The computer readable medium of claim 9, wherein the second storage area is a temporary data record in a temporary table.

16. (Original) The computer readable medium of claim 9, wherein the command is a positioned UPDATE and DELETE command in a relational database system supporting scrollable cursors and optimistic concurrency.

17. (Original) A relational database management system comprising:

data records stored on a plurality of data pages;

means for providing an identifier on each data page, the identifier indicating when any of the data records contained therein were last modified;

means for selecting a data record from a data page;

means for copying and storing the selected data record and the identifier from the data page to a second storage area;

means for determining a current identifier from the data page; and

means for verifying that the selected data record has not been modified since the time that it was copied to the second storage area by determining that the stored identifier is the same as the current identifier from the data page.

18. (Original) The system of claim 17, wherein the second storage area is a temporary data record in a temporary table.

19. (Original) The system of claim 17, wherein the relational database management system supports a positioned UPDATE and DELETE command and scrollable cursors and optimistic concurrency.

20. (Original) The system of claim 17, wherein the identifier comprises a log sequence number (LSN).

21. (Original) The system of claim 17, wherein the identifier comprises a time stamp.

22. (New) A method for optimizing command execution in a database system, wherein data records are stored on a plurality of data pages therein, the method comprising the steps of:

(a) providing an identifier to each data page, the identifier indicating when any of the data records contained therein were last modified;

(b) selecting a data record from a data page;

(c) copying the selected data record and copying and storing the identifier to a second storage area;

(d) verifying that the selected data record has not been modified since the time that it was copied to the second storage area based upon the identifier by:

(d1) determining a current identifier for the data page;

(d2) comparing the current identifier with the stored identifier, and

(d3) concluding the selected data record has not been modified when the current identifier is the same as the stored identifier; and

(e) executing the command.

23. (New) A computer readable medium containing programming instructions for optimizing command execution in a database system, wherein data records are stored on a plurality of data pages therein, the programming instructions for:

(a) providing an identifier to each data page, the identifier indicating when any of the data records contained therein were last modified;

(b) selecting a data record from a data page;

(c) copying the selected data record and copying and storing the identifier to a second storage area;

(d) verifying that the selected data record has not been modified since the time that it was copied to the second storage area based upon the identifier by:

(d1) determining a current identifier for the data page;

(d2) comparing the current identifier with the stored identifier, and (d3) concluding the selected data record has not been modified when the current identifier is the same as the stored identifier; and

(e) executing the command.

24. (New) A relational database management system comprising:

data records stored on a plurality of data pages;

means for providing an identifier on each data page, the identifier indicating when any of the data records contained therein were last modified;

means for selecting a data record from a data page;

means for copying and storing the selected data record and the identifier from the data page to a second storage area;

means for determining a current identifier from the data page; and

means for verifying that the selected data record has not been modified since the time that it was copied to the second storage area by comparing the identifier stored in the second storage area with the current identifier, and concluding that the selected data record has not been modified if the stored identifier is the same as the current identifier.

APPENDIX B

EVIDENCE

(NONE)

APPENDIX C

RELATED PROCEEDINGS

(NONE)